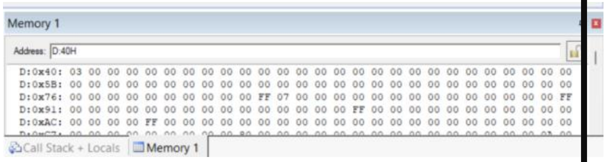
1.Find the number of 1’s in YOUR_ROLL_ NUMBER (decimal value) and store it in 40H

Ans:

```
MOV A,#21 ;load decimal value of 21 into accumulator

MOV R0,#00H ;register for counting no. of ones

MOV R1,#08H ;register for counting no. of bits

L2:RLC A ;rotate left with carry

JNC L1 ;jump if no carry to L1

INC R0 ;increment r0 if carry generated

L1:DJNZ R1,L2 ;bit counter decremented and jump to L2 til it reaches 0

MOV 40H,R0 ;display final result at 40h

END ;end
```



Memory 1

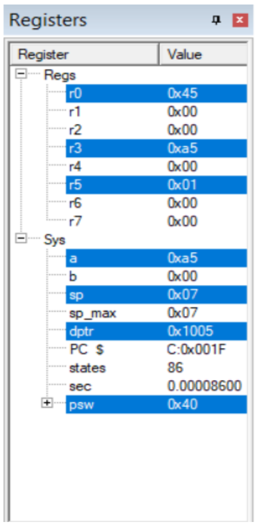
Address: D:40H

```
D:0x40: 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x5B: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
D:0x76: 00 00 00 00 00 00 00 00 00 00 FF 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF
D:0x91: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF 00 00 00 00 00 00 00 00 00 00 00
D:0xAC: 00 00 00 00 FF 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

2. Assume that external RAM locations 1000h–1004h have the values 7Eh,1Bh, YOUR_ROLL_NUMBER (decimal value), 5Fh, 8Ch respectively. Write a program to find the sum of the values and store the results - low byte in R3 and high byte in R5

Ans:

```
ORG 00H ;begin from origin

MOV DPTR,#1000H ;set value of DPTR as 1000h

MOV R1,#05H ;move 5h to r1

MOV R0,#40H ;move 40h to r0

HERE:MOV A,#00H ;move 0h to accumulator

MOVC A,@A+DPTR ;move the value that dptr points to a

MOV @R0,A ;move  content of accumulator to location pointed by r0

INC DPTR ;increment the value of DPTR

INC R0 ;incremet the value of r0

DJNZ R1,HERE ;decrement for non zero the value of r1 and jump to HERE

MOV R6,#05H ;move 5h to r6

MOV R0,#40H ;move 40h to r0

MOV R5,#00H ;move 0h to r5

CLR A ;clear the accumulator
```



Registers

Register	Value
Regs	
r0	0x45
r1	0x00
r2	0x00
r3	0xa5
r4	0x00
r5	0x01
r6	0x00
r7	0x00
Sys	
a	0xa5
b	0x00
sp	0x07
sp_max	0x07
dptr	0x1005
PC $	C:0x001F
states	86
sec	0.00008600
psw	0x40

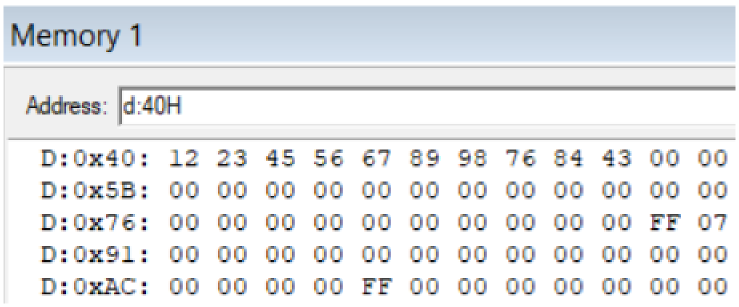
```
CLR C ;clear the carry

RPT: ADD A,@R0 ;add content of accumulator with value in location pointed by r0

JNC DOWN ;jump if no carry to down

INC R5 ;increment r5

DOWN:INC R0 ;increment r0

DJNZ R6,RPT ;decrement r6 and jump to RPT if not zero

MOV R3,A ;move content of A to r3

ORG 1000H ;begin from 1000h

STORE :DB 7Eh,1Bh,21h,5Fh, 8Ch ;storing values

END ;end
```

3. Ten hex numbers are stored in RAM location 40h onwards. Write a program to find the biggest number in the set. The biggest number should finally be saved in (35+ YOUR_ROLL_ NUMBER (decimal value))h

Ans:

```
ORG 00H ;begin from origin

MOV R6,#10H ;loop counter

MOV R0,#40H ;move 40h to r0

MOV 56H,@R0 ;move contents pointed by r0 to 56h

BACK: MOV A,@R0 ;move contents pointed by r0 to a

INC R0 ;increment r0

CJNE A,56H,L1 ;compare a and 56 h contents and jump to L1 if they are not equal

SJMP L2 ;short jump to L2

L1:JC L2 ;jump for carry to L2

MOV 56H,A ;move contents of a to 56h location

L2:DJNZ R6,BACK ;decrement and jump to back if r6 not equal zero

EXIT:SJMP EXIT

END ;end
```

input

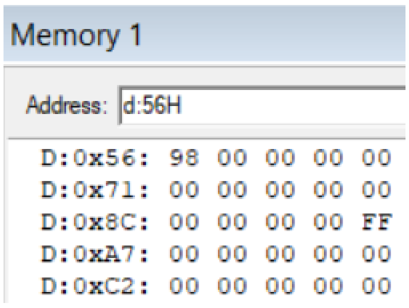


Memory 1

Address: d:40H

```
D:0x40: 12 23 45 56 67 89 98 76 84 43 00 00
D:0x5B: 00 00 00 00 00 00 00 00 00 00 00 00
D:0x76: 00 00 00 00 00 00 00 00 00 00 FF 07
D:0x91: 00 00 00 00 00 00 00 00 00 00 00 00
D:0xAC: 00 00 00 00 FF 00 00 00 00 00 00 00
```

output



Memory 1

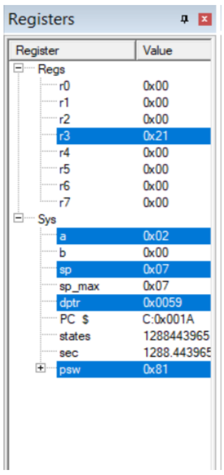
Address: d:56H

```
D:0x56: 98 00 00 00 00
D:0x71: 00 00 00 00 00
D:0x8C: 00 00 00 00 FF
D:0xA7: 00 00 00 00 00
D:0xC2: 00 00 00 00 00
```

4. Assume internal ROM memory location 50h – 59h has the data 16,23,37,32,44,59,65,62,2,29 (all decimal) which is the daily temperature for ten days. Search to see if any of the value equals YOUR_ROLL_ NUMBER(decimal value). If value does exist in the table, give its location to R3; Otherwise make R3 = YOUR_ROLL_ NUMBER (hexadecimal value)

Ans:

```
 MOV DPTR ,#50H ;getting the starting location of external data

MOV R1 ,#9H ; counter

BACK :MOV A ,#00H ; clearing A

MOVC A,@A+DPTR ; travesing external data

INC DPTR ; incrementing data pointer

CJNE A,#21,L1 ; comapring the values bw A and rollno

MOV A,#06H ; assign #06h to A

ADD A,R1 ;  adding A and counter for extracting the address of value

MOV R1,A ;  store result in r1

MOV A,#60H ; assign A with 60h

SUBB A,R1 ;  getting the address of value

MOV R3,A ; storing the address to A

STOP:SJMP STOP ; short jump

L1: MOV R3,#21H ; storing roll no to r3

DJNZ R1,BACK ; decrement counter

IT:SJMP IT ; short jump

ORG 50H ; origin statement from 50h

 DATA: DB 16,23,37,32,44,59,65,62,2,29 ; storing the values

 HERE: SJMP HERE ; short jump

 END ; end
```



Registers

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x21
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x02
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0059
PC $	C:0x001A
states	1288443965
sec	1288.443965
psw	0x81

5. Write PUSH instructions to push the contents of the registers on stack after the execution of the following set of instructions

MOV R0, # DATE_OF_BIRTH (decimal value)
MOV R1, #25h
MOV R5, #0FEh
MOV A, #0CDh
MOV B, # YOUR_ROLL_ NUMBERh
Write down the contents of stack after execution. Mention each stack locations address and its contents as a table

Ans:MOV R0,#14 ;move 14 to r0

MOV R1,#25H ;move 25h to r1

MOV R5,#0FEH ;move 0FEH to r5

MOV A,#0CDH ;move 0CDH to accumulator

MOV B,#21H ;move 21H to b register b

PUSH 00 ;push ro to stack

PUSH 01 ;push r1 to stack

PUSH 05 ;push r5 to stack

PUSH 0E0H ;push register a to stack

PUSH 0F0H ;push register b to stack

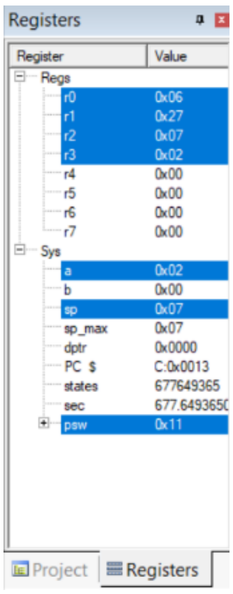
END ;end

Address	Data
0X000B	0E
0X000D	25
0X000F	FE
0X0011	CD
0X0013	21

6. Write instructions to use the registers of bank 2, and load the value 06h in the register R0. Add YOUR_ROLL_ NUMBER to R0 and load that value in R1.Extract the lower and upper byte of the value stored in R1 and store them in R2 and R3. Show the contents of the respective RAM locations corresponding to bank 2 registers after executing the instructions.

Ans:

```
SETB PSW.4 ;selecting the register bank 2

CLR PSW.3 ;selecting the register bank 2

MOV R0,#06H ;moving #06h to ro

MOV A,#21H ; moving roll no. to Accumulator

ADD A,R0  ; adding both

MOV R1 ,A ; storing the result to r1

MOV A ,#0FH ;moving #ofh to A for extraction

ANL A,R1 ; and logic

MOV R2 ,A ; storing result to r2

MOV A ,#0F0H ; moving #ofoh to A

ANL A,R1 ; and logic for extraction

SWAP A ; swapping the nibble

MOV R3,A ; storing te result in r3

STOP: SJMP STOP ; short jump

END ; end statement
```



Registers

Register	Value
Regs	
r0	0x06
r1	0x27
r2	0x07
r3	0x02
r4	0x00
r5	0x00
r6	0x00
r7	0x00
Sys	
a	0x02
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0000
PC $	C:0x0013
states	677649365
sec	677.6493650
psw	0x11

7. Write instructions to use assembler directives to place constants 23, 45, 56h, A4h and character string “YOUR_NAME” in consecutive program memory locations beginning from 100h. Show the contents of the memory after execution as in table of Question 5

Ans:

```
ORG 100H ;begin from origin

DATA1: DB 23 ;define bit as 23

DATA2: DB 45 ;define bit as 45

DATA3: DB 56H  ;define bit as 56h

DATA4: DB 'A4H'  ;define bit as A4H

DATA5: DB 'Basil Roy' ;define bit as basilroy

END ;end
```

Address	Data
100H	17
101H	2D
102H	56
103H	41
104H	34
105H	48
106H	42
107H	61

108H	73
109H	69
10AH	6C
10BH	20
10CH	52
10DH	6F
10EH	79

8. Assume that 7 BCD data items are stored in RAM locations starting at 50h, as shown below. Write a program to find the sum of all the numbers. The result must be in BCD

50 = (23),51= (38),52 =(65),53=(84),54 =(91),55=(29),56=(45)

Ans:

ORG 0 ;begin at 00h

MOV R0,#50H ;to make ro as pointer

MOV R4,#07H ;loop counter

MOV R5,#00 ;to store upper byte of result

MOV R6,#00 ;to store lower byte of result

CLR A ;clear accumulator

NEXT:ADD A,@R0 ;add contents from 5oh onwards

DA A ;decimal adjusting accumulator

JNC SKIP ;jump if no carry to skip

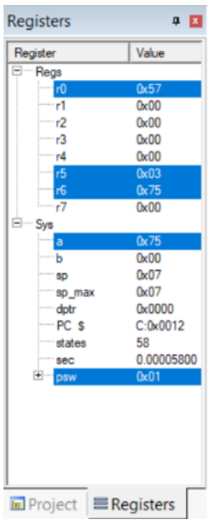
INC R5 ;increment r5 if carry generated

SKIP:INC R0 ;inrement pointer by 1

DJNZ R4,NEXT ;decrement and jump if nin zero from r4 to next

MOV R6,A ;copy lower byte from accumulator to r6

END ;end



Register	Value
Regs	
r0	0x57
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x03
r6	0x75
r7	0x00
Sys	
a	0x75
b	0x00
sp	0x07
sp_max	0x07
dptr	0x0000
PC $	C:0x0012
states	58
sec	0.00005800
psw	0x01

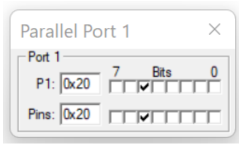
9. Two numbers are stored in registers R0 and R1 of register bank 1 of which one of them is YOUR_ROLL_NUMBER and the other is your DATE_OF_BIRTH.Write a program to verify if their sum is odd or even. If the sum is an odd number, send a high pulse to P1.5 else a low pulse to P1.2. Keep the screen shot of the port pin contents along with your program.

Ans:

SETB PSW.3 ;selecting bank 1

CLR PSW.4 ;selecting bank 1

MOV R0,#21 ;store roll no. to r0



P1: 0x20 ☐ ☐ ✓ ☐ ☐ ☐ ☐ ☐

Pins: 0x20 ☐ ☐ ✓ ☐ ☐ ☐ ☐ ☐

```
MOV R1,#14 ;store dob to r1

MOV A,R0 ;copy contents of r0 to accumulator

ADD A,R1 ;add contents of r1 with accumulator

MOV P1,#00H ;making port 1 bits 0

CLR C ;clear carry

RRC A ;rotate right with carry in A

JC LOOP1 ;jump for carry to loop 1

CLR P1.2 ;send 0 to pin p1.2 if even

SJMP HERE ;short jumb to here

LOOP1:SETB P1.5 ;make pin p1.5 1 if odd

HERE:SJMP HERE

END ;end
```

10. Using Edsim Simulator posted along with this assignment, display your YOUR_ROLL_NUMBER and DATE_OF_BIRTH alternatively on the 7-segment display with a delay in between them.

Ans:

```
ORG 00h ;begin from origin

INITIAL:MOV R3,#02h     ;loop counter

MOV DPTR,#0200h ;move value 0200h to data pointer

BACK:CLR A ;clear accumulator

SETB P0.7 ;enable decoder

SETB P3.3 ;selecting first(from left)seven segment display

SETB P3.4 ;selecting first(from left)seven segment display

MOVC A,@A+DPTR ;moving contents of location pointed by dptr to a

MOV P1,A ;move contents of a to p1

ACALL DELAY ;calling subroutine delay

INC DPTR ;increment dptr

SETB P0.7 ;enable decoder

CLR P3.3 ;selecting second(from left)seven segment display

SETB P3.4 ;selecting second(from left)seven segment display

CLR a ;clear accumulator

MOVC A,@A+DPTR ;moving contents of location pointed by dptr to a
```

```
MOV P1,A ;move contents of a to p1

ACALL DELAY ;calling subroutine delay

INC DPTR ;increment dptr

DJNZ R3,BACK ;decrement r3 and jump to back until r3 becomes 0

DELAY:MOV R0, #05h  ; Outer loop

AGAIN:MOV R1, #0BCh  ; inner loop

HERE:DJNZ R1, HERE   ;  decrement and jump if r1 not 0

DJNZ R0, AGAIN  ; decrement and jump if r0 not 0

RET

ORG 0200h ;begin from 0200h

DATAS:DB   0A4h,0F9h,0F9h, 099h ;storing roll no and dob

END ;end
```

Ootput: